Java mocking frameworks

jMock jmock.org

EasyMock easymock.org

Mockito code.google.com/p/mockito

All examples based on JUnit.

Verify interactions

```
import static org.mockito.Mockito.*;

//mock creation:
List mockedList = mock(List.class);

//using mock object - doesn't throw any "unexpected interaction" exception:
mockedList.add("one");
mockedList.clear();

//selective & explicit verification:
verify(mockedList).add("one");
verify(mockedList).clear();
```

Stub method calls

```
//You can mock concrete classes, not only interfaces
LinkedList mockedList = mock(LinkedList.class);

//stubbing - before execution
when(mockedList.get(0)).thenReturn("first");

//following prints "first"
System.out.println(mockedList.get(0));

//following prints "null" because get(999) was not stubbed
System.out.println(mockedList.get(999));
```

Stub exceptions

```
LinkedList mockedList = mock(LinkedList.class);


//stubbing

when(mockedList.get(0)).thenReturn("first");

when(mockedList.get(1)).thenThrow(new RuntimeException());

doThrow(new RuntimeException()).when(mockedList).clear();


//following prints "first"

System.out.println(mockedList.get(0));


//following throw runtime exception

mockedList.get(1);

mockedList.clear();
```

Argument matching

```
//stubbing using built-in anyInt() argument matcher
when(mockedList.get(anyInt())).thenReturn("element");

//stubbing using hamcrest (let's say isValid() returns your own hamcrest matcher):
when(mockedList.contains(argThat(isValid()))).thenReturn("element");

//following prints "element"
System.out.println(mockedList.get(999));

//you can also verify using an argument matcher
verify(mockedList).get(anyInt());
```

```
verify(mock).someMethod(anyInt(), anyString(), eq("third argument"));
//above is correct - eq() is also an argument matcher

verify(mock).someMethod(anyInt(), anyString(), "third argument");
//above is incorrect - exception will be thrown because third argument is given without an
argument matcher.
```

Custom Matchers

Annotations

```
public class MockitoTest  {

 @Mock MyDatabase databaseMock;

  @Before
  protected void setUp() throws Exception {
    MockitoAnnotations.initMocks(this);
  }

  @Test
  public void testQuery()  {
   ClassToTest t  = new ClassToTest(databaseMock);

   boolean check = t.query("* from t");
   assertTrue(check);
  Mockito.verify(mock).query("* from t");
  }
}
```

Annotations

```
@RunWith(MockitoJUnitRunner.class)
public class AnnotationMockTest {

    @InjectMocks
    private UserManager userManager;

    @Mock
    private UserService userService;
}
```

Spy